International Journal of Research and Applications
Oct - Dec © 2014 Transactions; 1(4): 144-148

ISSN (online): 2349-0020
http://ijraonline.com

ELECTRONICS ORIGINAL ARTICLE

CLAA, CSLA and PPA based Shift and Add Multiplier for General Purpose Processor

A. Sowjanya [1] and R. Mahalaxmi [2]
[1] M.Tech(Pursuing) and [2] Assistant Professor
[1 2] Department of ECE, Malla Reddy engineering college for Women,
Affiliated to Jawarlal Nehru Technological University, Dhulapally, Secunderabad - 506 014.
[1] sowji.sowjanya045@gmail.com and [2] mahalaxmi.engg@gmail.com

ABSTRACT

This project deals with the comparison of the VLSI design of the carry look-ahead adder (CLAA) based 32-bit unsigned integer multiplier and the VLSI design of the carry select adder (CSLA) based 32-bit unsigned integer multiplier. Both the VLSI design of multiplier multiplies two 32-bit unsigned integer values and gives a product term of 64-bit values. The CLAA based multiplier and CSLA based multiplier uses nearly the same delay time for multiplication operation. But the area needed for CSLA multiplier is less by the CLLA based multiplier to complete the multiplication operation.PPA is used to increase the speed then CSLA and CLAA.

Keywords: CLAA, CSLA, PPA, Delay, Area, Array Multiplier, VHDL Modeling & Simulation

1. INTRODUCTION

Digital computer arithmetic is an aspect of logic design with the objective of developing appropriate algorithms in order to achieve an efficient utilization of the available hardware. The basic operations are addition, subtraction, multiplication and division. In this, we are going to deal with the operation of additions implemented to the operation of multiplication. The repeated form of the addition operations and shifting results in the multiplication operations.

Given that the hardware can only perform a relatively simple and primitive set of Boolean operations, arithmetic operations are based on a hierarchy of operations that are built upon the simple ones. In VLSI designs, speed, power and chip area are the most often used measures for determining the performance and efficiency of the VLSI architecture.

Multiplications and additions are most widely and more often used arithmetic computations performed in all digital signal processing applications. Addition is a fundamental operation for any digital multiplication. A fast, area efficient and accurate operation of a digital system is greatly influenced by the performance of the resident adders. Adders are also very important component in digital systems because of their extensive use in these systems.

In this project we are going to compare the performance of different adders implemented to the multipliers based on area and time needed for calculation. On comparison with the carry look-ahead adder (CLAA) based multiplier the area of calculation of the carry select adder (CSLA) based multiplier takes more area and having same delay time. Here we are dealing with the comparison in the bit range of n*n (32*32) as input and 2n (64) bit output. Hence, to design a better architecture the basic adder blocks must have reduced delay time consumption and area efficient architectures. The demand is of DSP style systems for both less delay time and less area requirement for designing the systems.

Our interest is in the basic building blocks of arithmetic circuits that dominate in DSP applications, VLSI architectures, computer applications and where ever reduced area computation is needed.

2. CARRYLOOK AHEAD ADDER

Carry Look Ahead Adder can produce carries faster due to parallel generation of the carry bits by using additional circuitry. This technique uses calculation

of carry signals in advance, based on input signals. The result is reduced carry propagation time. For example, ripple adders are slower but use the least energy.

Let Gi is the carry generate function and Pi be the carry propagate function, Then we can rewrite the carry function as follows:

Gi = Ai . Bi (1)
Pi = (Ai xor Bi) (2)
Si = Pi xor Ci. (3)
Ci+1= Gi + Pi.Ci. (4)

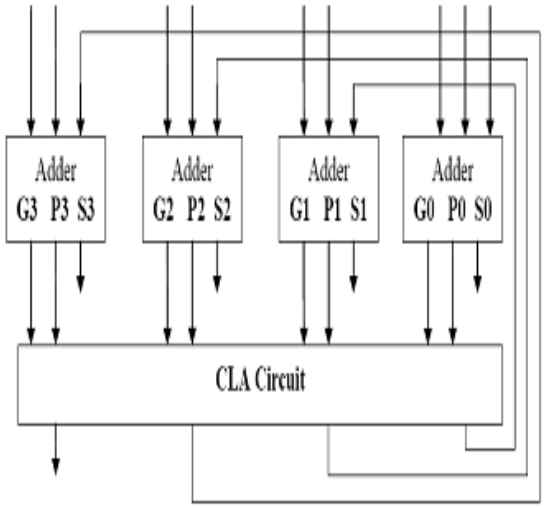

Figure 1. *Carry Look Ahead Adder*

Thus, for 4-bit adder, we can compute the carry for all the stages as shown below:

C1 =G0 + P0.C0 (5)
C2 = *G1* + P1.C1 = G1 + P1.G0 + P1.P0.C0 (6)
C3 = *G2* + P2.C2= G + P2.G1 + P2.P 1.G0 + P2.P1.P0.C0 (7)
C4 = G3 + P3.C3 = G3 + P3.G2 + P3.P2 .G1 + P3.P2.P1.G0 + P3.P2.P1.P0.C0 (8)
In general, we can write:
The sum function:
SUMi = Ai xor Bi xor Ci = Pi xor Ci (9)
The carry function: Ci+l= Gi + Pi.Ci. (10)

3. CARRY SELECT ADDER

The concept of CSLA is to compute alternative results in parallel and subsequently selecting the correct result with single or multiple stage hierarchical techniques. In CSLA both sum and carry bits are calculated for two alternatives Cin=O and 1. Once Cin is delivered, the correct computation is chosen using a mux to produce the desired output. Instead of waiting for Cin to calculate the sum, the sum is correctly output as soon as Cin gets there. The time taken to compute the sum is then avoided which results in good improvement in speed.

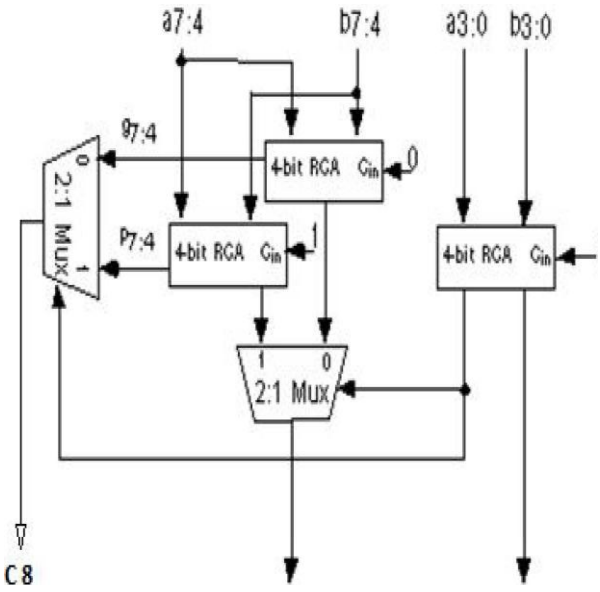

Figure 2. *Carry Select Adder*

In general, we can write the algorithm as:

If Carry in =1, then the sum and carry out are given by,

Sum (i) =a (i) xor b (i) xor '1' (11)

Carry (i+ 1) = (a (i) and b (i)) or (b (i) or a (i)) (12)

If Carry in =0, then the sum and carry out are given by,

Sum (i) = a (i) xor b (i). (13)
Carry (i+ 1) = (a (i) and b (i)). (14)
The sum function:
$Si = CiSi^0 + CiSi^1$ (15)
The carry function:
$Ci+1 = CiCi+1^0 + CiCi+1^1$ (16)

4. PARALLEL PREFIX ADDER

Parallel-prefix adders compute addition in two steps: one to obtain the carry at each bit, with the next to compute the sum bit based on the carry bit. Unfortunately, prefix trees are algorithmically slower than fast logarithmic adders, such as the carry propagate adders, however, their regular structures promote excellent results when compared to traditional CLA adders. This happens within VLSI architectures because a carry-look ahead adder, such as the one implemented in one of Motorola's processors, tends to implement the carry chain in the vertical direction instead of a horizontal one, which has a tendency to increase both wire density and fan-in/out dependence. Therefore, although logarithmic adder structures are one of the fastest adders algorithmically, the speed efficiency of the carry-look ahead adder has been hampered by diminishing returns given the fan-in and fan-out dependencies as well as the heavy wire load distribution in the vertical path. In fact, a traditional carry-look ahead adder implemented in VLSI can actually be slower than traditional linear-based adders, such as the Manchester carry adder.

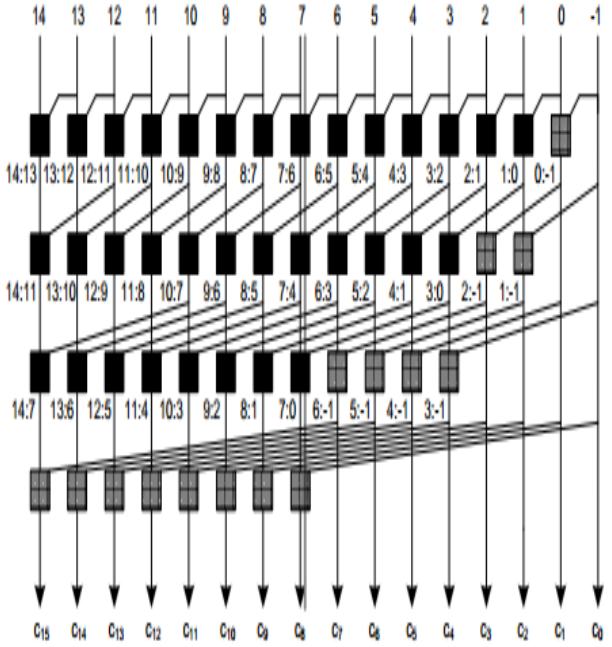

Figure 3. *16-bit Kogge-Stone Prefix Tree*

5. MULTIPLTER FOR UNSIGNED DATA

Multiplication involves the generation of partial products, one for each digit in the multiplier, as in Figure3.These partial products are then summed to produce the final product. The multiplication of two n-bit binary integers results in a product of up to 2n bits in length [2].

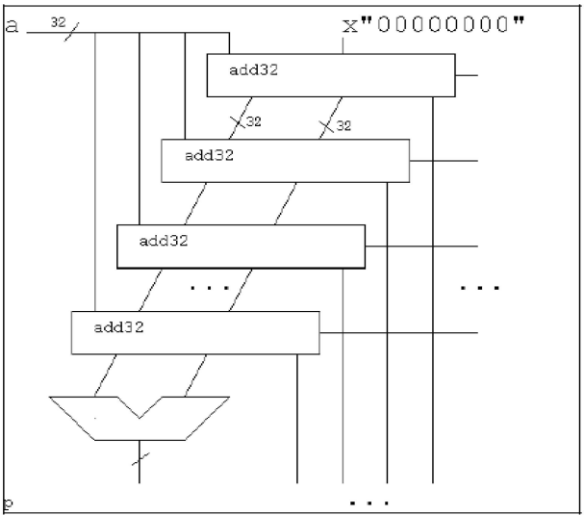

Figure 4. *A partial schematic of the multiplier*

We used the following algorithm to implement the multiplication operation for unsigned data.

6. MULTIPLICATION ALGORITHM

Let the product register size be 64 bits. Let the multiplicand registers size be 32 bits. Store the multiplier in the least significant half of the product register. Clear the most significant half of the product register.

Repeat the following steps for 32 times:

1. If the least significant bit of the product register is "1" then add the multiplicand to the most significant half of the product register.
2. Shift the content of the product register one bit to the right (ignore the shifted-out bit.)
3. Shift-in the carry bit into the most significant bit of the product register.

Figure 4.Shows a block diagram for such a multiplier [2].

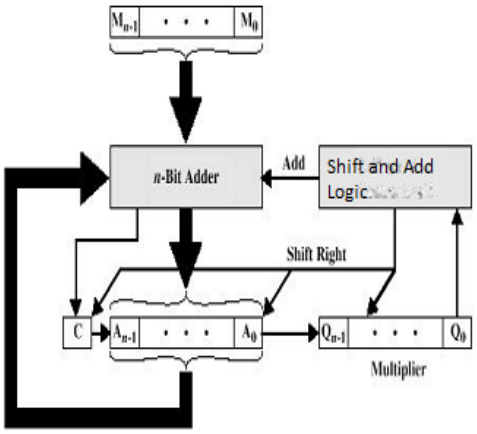

Figure 5. *Multiplier of two n-bit values.*

7. Simulation Results

The VHDL simulation of the two multipliers is presented in this section. In this, waveforms, timing diagrams and the design summary for both the CLAA and CSLA based multipliers are shown in the figures. The VHDL code for both multipliers, using CLAA and CSLA, are generated. The VHDL model has been developed using Xilinx ISE 13.2. The multipliers use two 32-bit values.

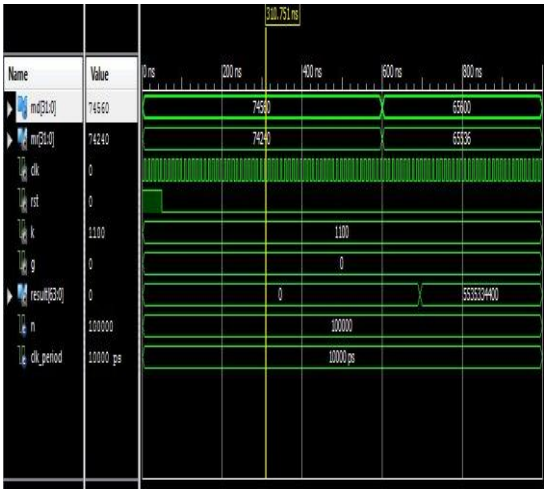

Figure 6. *Waveform of Carry look Ahead Based multiplier*

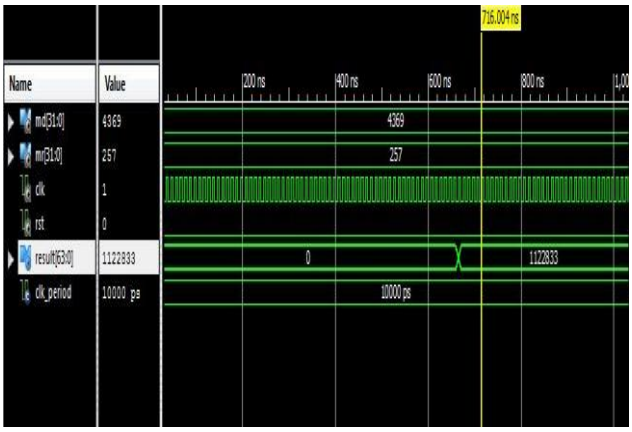

Figure 7. *Waveform of Carry Select Adder Based multiplier*

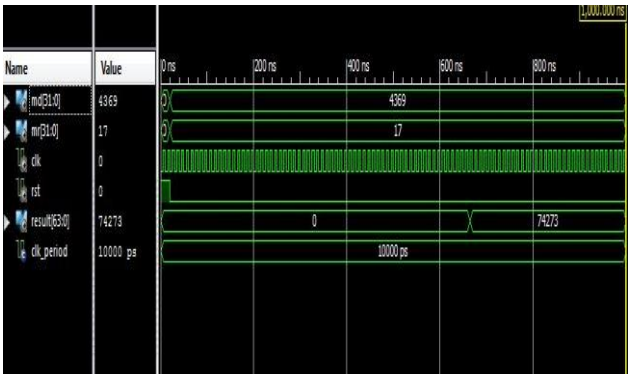

Figure 8. *Waveform of Parallel Prefix Adder Based multiplier*

8. Conclusion

A design and implementation of a VHDL-based 32-bit unsigned multiplier with CLAA and CSLA was presented. VHDL, a Very High Speed Integrated Circuit Hardware Description Language, was used to model and simulate our multiplier. Using CSLA improves the overall performance of the multiplier.

Thus a 31 % area delay product reduction is possible with the use of the CSLA based 32 bit unsigned parallel multiplier than CLAA based 32 bit unsigned parallel multiplier.

9. Future Work

This 32 bit multiplier can be further extended to 64 bit multiplier and 128 bit multiplier using the proposed method for multiplication operation can be done as future work.

REFERENCES

[1]. P. Asadi and K. Navi, "A novel highs-speed 54-54 bit multiplier", Am. J Applied Sci., vol. 4 (9), pp. 666-672. 2007.

[2]. W. Stallings, Computer Organization and Architecture Designing for Peljormance, 71h ed., Prentice Hall, Pearson Education International, USA, 2006, ISBN: 0-13-185644-8.

[3]. F. Wakerly, Digital Design-Principles and Practices, 4th ed., Pearson Prentice Hall, USA, 2006. ISBN: 0131733494.

[4]. A. Sertbas and R.S. Ozbey, "A performance analysis of classified binary adder architectures and the VHDL simulations", J Elect. Electron. Eng., Istanbul, Turkey, vol. 4, pp. 1025-1030,2004.

[5]. P. S. Mohanty, "Design and Implementation of Faster and Low Power Multipliers", Bachelor Thesis. National Institute of Technology, Rourkela, 2009.

[6]. S. Brown and Z. Vranesic, Fundamentals of Digital Logic with VHDL Design, 2nd ed., McGraw-Hill Higher Education, USA, 2005. ISBN: 0072499389.

[7]. J. R. Armstrong and F.G. Gray, VHDL Design Representation and Synthesis, 2nd ed. , Prentice Hall, USA, 2000. ISBN: 0-13-021670-4.

[8]. Z. Navabi, VHDL Modular Design and Synthesis of Cores and Systems, 3rd ed., McGraw-Hill Professional, USA, 2007. ISBN: 9780071508926.

[9]. P. C. H. Meier, R. A. Rutenbar and L. R. Carley, "Exploring Multiplier Architecture and Layout for low Power", CIC'96, 1996.

[10]. Software Simulation Package: Direct VHDL, Version 1.2, 2007, Green Mounting Computing Systems, Inc., Essex, VT, UK.

[11]. Hasan Krad and Aws Yousi("Design and Implementation of a Fast Unsigned 32-bit Multiplier Using VHDL", 2010.

AUTHOR'S BIOGRAPHY

1. A. Sowjanya is pursuing her M.Tech from Malla Reddy engineering college for Women, Hyderabad in the department of Electronics & Communications Engineering with specialization in embedded systems.

2. R.Mahalaxmi is working as an Assistant Professor in the department of Electronics & Communication Engineering in Malla Reddy engineering college for Women, Hyderabad. She completed masters from PRRM Engineering College. (She has over 07 years of teaching experience.)